Ludwig-Maximilians-Universität München
Institut für Informatik
Prof. Dr. Matthias Schubert
Sebastian Schmoll, Sabrina Friedl

Munich, 29.10.2018

Deep Learning and Artificial Intelligence

WS 2018/19

Exercise 2: Math Primer

Exercise 2-1 Jacobian Matrices

Suppose $F : \mathbb{R}^d \to \mathbb{R}^n$ is a vector-valued function that maps an input vector $x \in \mathbb{R}^d$ to an output vector $F(x) \in \mathbb{R}^n$. The Jacobian matrix J_F is defined as:

$$J_F := \begin{pmatrix} \frac{\partial F}{\partial x_1} & \cdots & \frac{\partial F}{\partial x_d} \end{pmatrix} = \begin{pmatrix} \frac{\partial F_1}{\partial x_1} & \cdots & \frac{\partial F_1}{\partial x_d} \\ \vdots & \ddots & \vdots \\ \frac{\partial F_m}{\partial x_1} & \cdots & \frac{\partial F_m}{\partial x_d} \end{pmatrix} \in \mathbb{R}^{n \times d},$$

i.e. it contains the derivatives of each output with regard to each input: $(J_F)_{ij} = \frac{\partial F_i}{\partial x_j}$.

In the following, let $x \in \mathbb{R}^d$ be a vector with d elements and $W \in \mathbb{R}^{n \times d}$ be a matrix with n rows and d columns.

(a) Given $z = Wx$. Calculate the Jacobian matrix $J_z = \frac{\partial z}{\partial x}$.

(b) Given $z = x^T W^T$. Calculate the Jacobian matrix $J_z = \frac{\partial z}{\partial x}$.

(c) Given $z = f(x)$, where f is applied elementwise to the vector x, i.e. $z_i = f(x_i)$.
Calculate the Jacobian matrix $J_z = \frac{\partial z}{\partial x}$ (not the gradient $\nabla f(x)$).

(d) Given $z = Wx$ and a loss function $L : \mathbb{R}^n \to \mathbb{R}$ that maps z to a scalar loss $L(z)$.
Calculate $\frac{\partial L}{\partial W} = \frac{\partial L}{\partial z} \frac{\partial z}{\partial W}$ (chain rule).

Exercise 2-2 Softmax and Cross-Entropy Loss

(a) Given $\hat{y} = softmax(z)$ with $\hat{y}_i = \frac{e^{z_i}}{\sum_{k=1}^N e^{z_k}}$, where $\hat{y} \in \mathbb{R}^N$ and N is the number of classes of a classification problem. Calculate $\frac{\partial \hat{y}_i}{\partial z_j}$.

(b) Given $\hat{y} = softmax(z)$, a target vector $y \in \mathbb{R}^N$ and the cross-entropy loss function defined as

$$L(y, \hat{y}) = -\sum_{k=1}^N y_k \log \hat{y}_k.$$

Calculate $\frac{\partial L}{\partial z_i}$. *Hint:* Make use of the chain rule $\frac{\partial L}{\partial z_j} = \sum_i \left(\frac{\partial L}{\partial \hat{y}_i} \frac{\partial \hat{y}_i}{\partial z_j} + \frac{\partial L}{\partial y_i} \frac{\partial y_i}{\partial z_j} \right)$. Moreover, reuse the results of exercise 2a) and the fact that vector y contains the probabilties for each class i that sum up to one, i.e: $\sum_i y_i = 1$.

Exercise 2-3 Mean Squared Error

Consider the input dataset $X \in \mathbb{R}^{n \times d}$ with n samples of size d, a target vector $y \in \mathbb{R}^n$, a weight vector $w \in \mathbb{R}^d$ and a prediction $\hat{y} = Xw$. The mean squared error (MSE) is defined as the sum over the squared differences between the prediction $\hat{y}_i$ and the true values y_i for each instance:

$$L(y, \hat{y}) = \frac{1}{n} \sum_{i=1}^{n} (\hat{y}_i - y_i)^2,$$

where $\hat{y}_i = w^T x_i$ and $x_i \in \mathbb{R}^d$ is one sample of the dataset (corresponding to one row in X).

Find the vector w that minimizes the MSE loss function!

Hint: You can write the sum above as a vector product! Moreover, you can use the following identities: $(Ax)^T = x^T A^T$, $\frac{\partial x^T A x}{\partial x} = 2Ax$ and $\frac{\partial x^T b}{\partial x} = b$ for vectors x and b and matrices A.